AI SHOP ASSISTANT

Chat System

3rd Sept 2025

Owner

Niranjan Singh

Introduction

Our project aims to build a chat boat. Online shopping has become the go-to option for many consumers. However, the overwhelming number of choices and the lack of personalized assistance can make the shopping experience daunting. To address this, we have developed **ShopAssist AI, a chatbot that combines the power of large language models and rule-based functions to ensure accurate and reliable information delivery**.

Background Project

Case Study:

Given a dataset containing information about laptops (product names, specifications, descriptions, etc.), build a chatbot that parses the dataset and provides accurate laptop recommendations based on user requirements*.

Purpose

The purpose of this document is to describe the architecture and design of a conversational AI platform that helps users interact through natural language to obtain relevant information, recommendations, or product matches.

The system is designed to be:

- Modular and extensible
- Secure and observable
- Capable of handling structured and unstructured user input

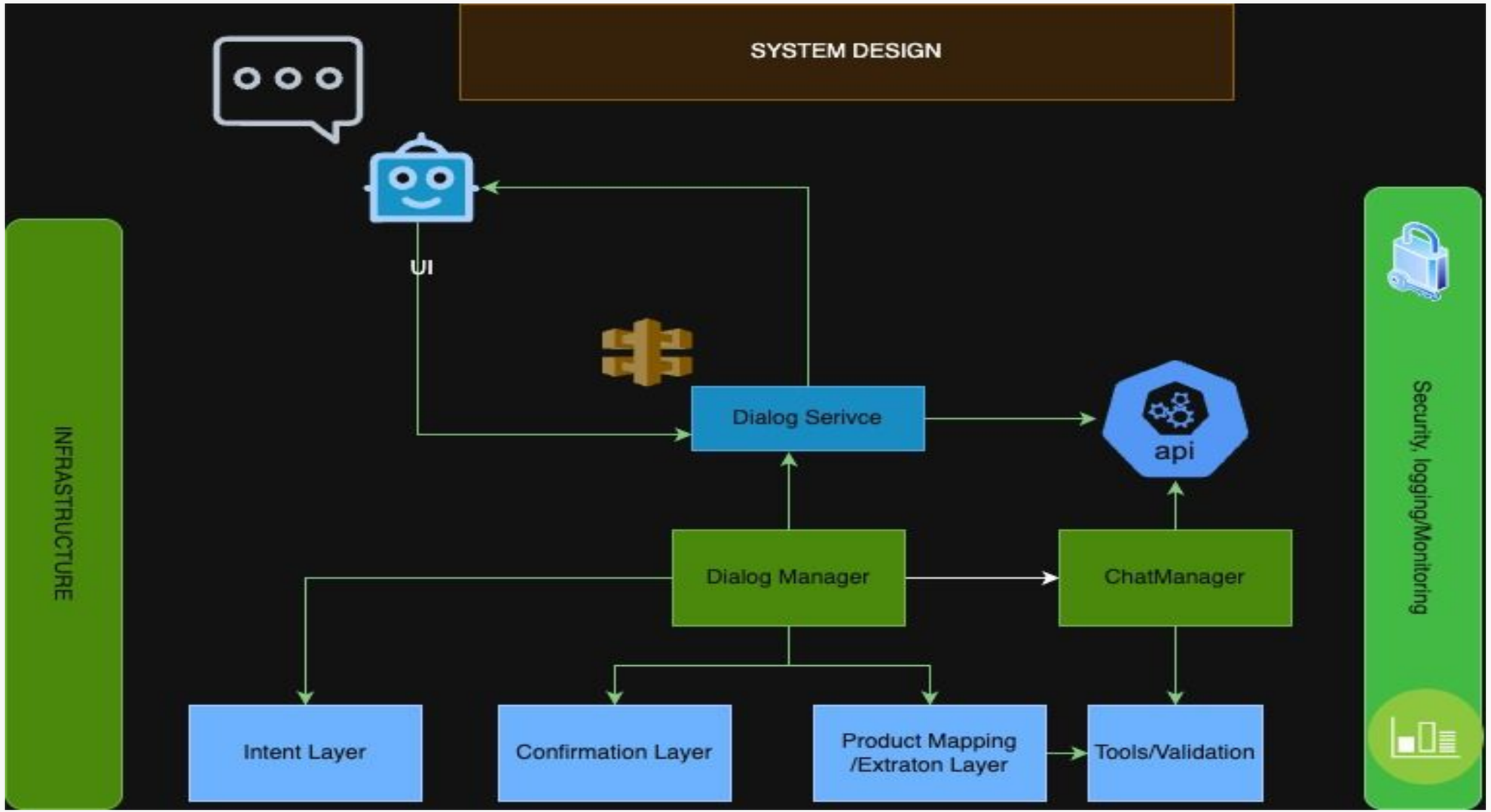
Scope

The system enables end-users to communicate via a chatbot UI, process their inputs through a dialog management pipeline, validate and map inputs to structured formats, and integrate with backend APIs for recommendations or validations.

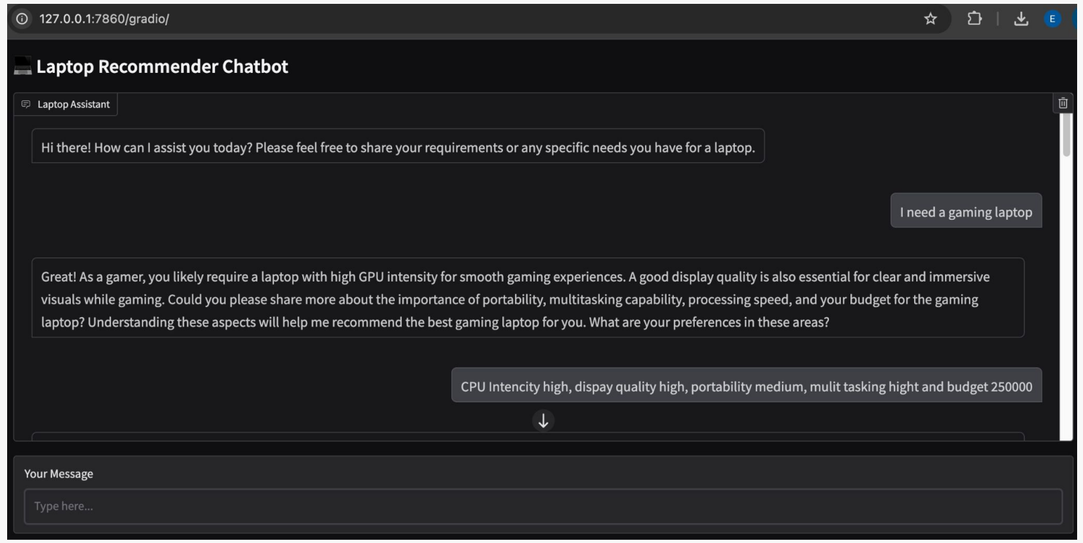
Use cases include:

- Product recommendation (e.g., laptops, devices)
- Information queries and confirmations
- Workflow automation via dialog orchestration

System Architecture



System Components



User Interface (UI)

- Entry point for user interactions.
- Can be web, mobile, or embedded chat widget.
- Captures user input and displays chatbot responses.

Dialog Service

- Middleware responsible for routing messages between the UI and backend components.
- Ensures reliable delivery of user input to the Dialog Manager.
- Can support scaling and load balancing.

System Components

Processing Layers

Intent Layer

- Extracts user intent (e.g., "Find laptop", "Confirm requirements").
- Uses NLP or ML models.

Confirmation Layer

- Validates extracted inputs (e.g., budget, GPU specs).
- Triggers clarifying questions if input is missing or inconsistent.

Product Mapping / Extraction Layer

- Maps user preferences to structured data.
- Converts natural language into standard attributes (e.g., GPU intensity: high).
- Interfaces with product catalogs or knowledge bases.

Tools / Validation Layer

- Applies business logic and rules.
- Ensures values match expected formats (e.g., budget is numeric, categories are valid).

System Components

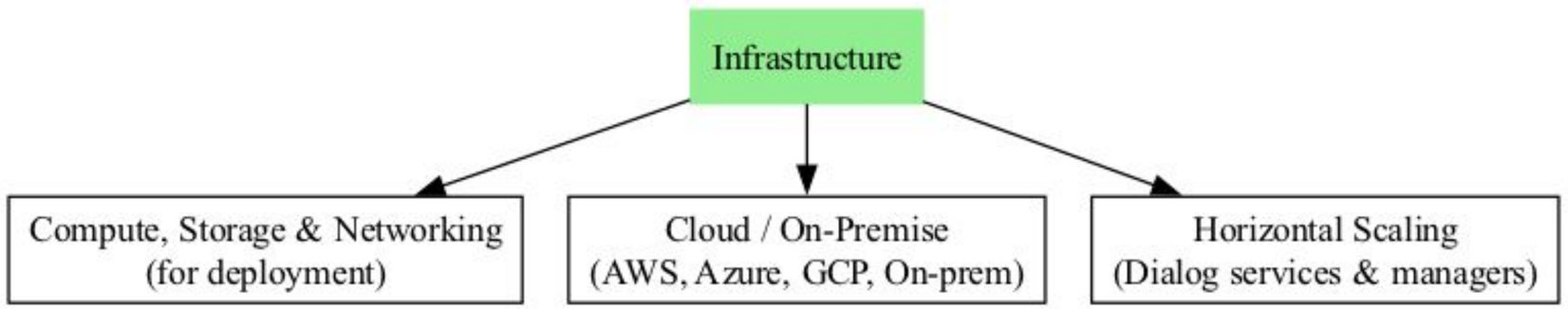
ChatManager

- Handles backend orchestration and API calls.
- Responsibilities:
  - Connect with product databases and recommendation services.
  - Validate extracted data.
  - Return structured responses to Dialog Manager.

System Components

Infrastructure

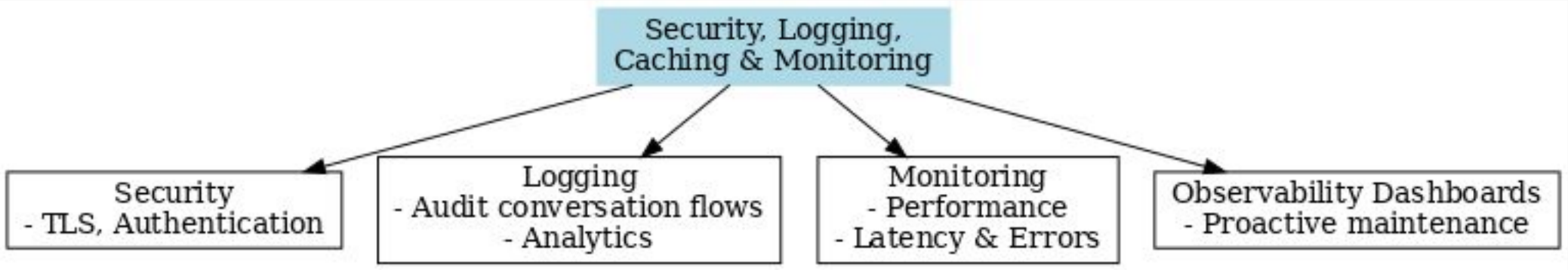
- Provides compute, storage, and networking for deployment.
- Can be cloud-based (AWS, Azure, GCP) or on-premise.
- Supports horizontal scaling of dialog services and managers.



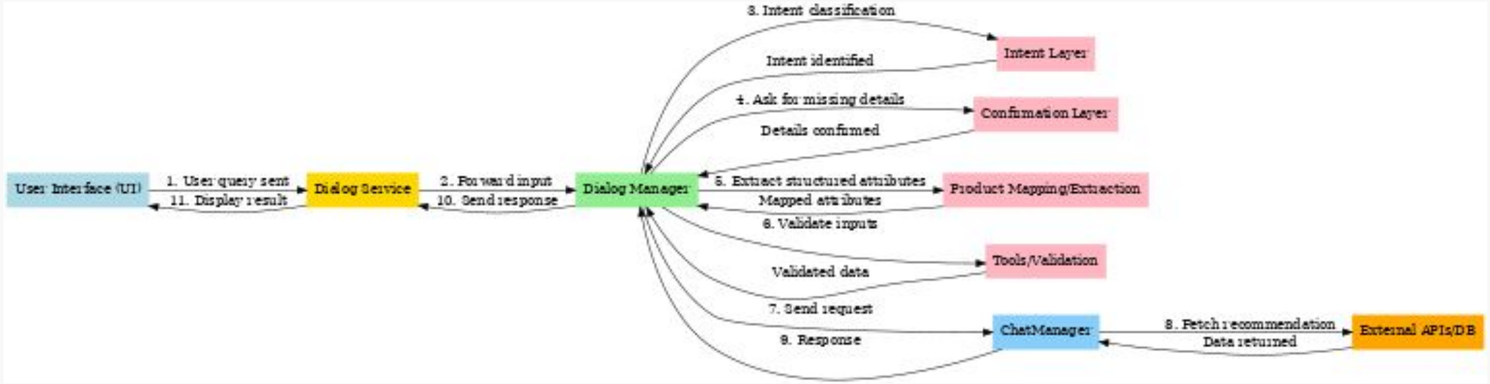
System Components

Security, Logging,Caching, and Monitoring

- Ensures secure communication between components (TLS, authentication).
- Logs all conversation flows for auditing and analytics.
- Monitors system performance, latency, and error rates.
- Provides observability dashboards for proactive maintenance.



System Components



Sequence Flow

1. User submits query via UI.
2. Dialog Service receives and forwards input to Dialog Manager.
3. Dialog Manager invokes Intent Layer to classify user request.
4. If details are incomplete, Confirmation Layer asks clarifying questions.
5. Valid input is sent to Product Mapping Layer for structured extraction.
6. Extracted attributes are validated by Tools/Validation Layer.
7. ChatManager invokes APIs or databases to fetch recommendations.
8. Response flows back through Dialog Manager → Dialog Service → UI.
9. All interactions are logged and monitored by Security/Logging/Monitoring.

System Components

Security Considerations

- Authentication & Authorization for API calls.
- Input validation & sanitization to prevent injection attacks.
- Data privacy for user conversations.
- End-to-end encryption for communication channels.
- Role-based access control (RBAC) for monitoring and admin interfaces.

System Components

Non-Functional Requirements

- Scalability: Handle thousands of concurrent users.
- Resilience: Retry and fallback mechanisms in dialog service.
- Low Latency: Real-time conversational response (<1s ideally).
- Extensibility: Easy addition of new intents, validation tools, and product categories.
- Observability: End-to-end logging, tracing, and dashboards.

System Components

Key Benefits

- Layered modular design ensures maintainability.
- Dialog Manager as orchestrator simplifies flow control.
- Validation + Mapping ensures data integrity and accurate recommendations.
- Security & monitoring baked in from the start.

Project File Structure

📂 Project Root

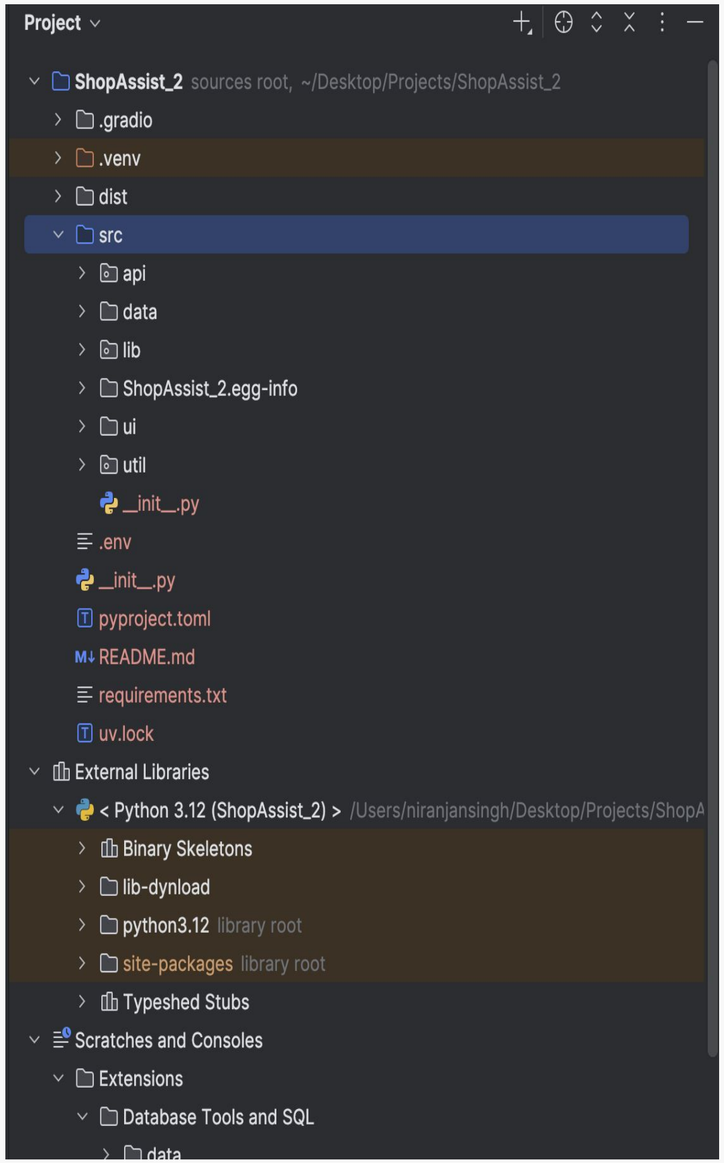
- ShopAssist_2 → Main project folder.
- .gradio/ → Likely related to Gradio UI setup/configuration (for chatbot interface).
- .venv/ → Local Python virtual environment (isolates dependencies).
- dist/ → Distribution folder (probably used for packaging your project).

📂 src/ (Main Source Code)

- api/ → Handles API-related logic (backend services, endpoints).
- data/ → Data storage, datasets, or configuration files used by the app.
- lib/ → Utility libraries or third-party integrations.
- ShopAssist_2.egg-info/ → Metadata about the installed package (auto-generated when installing in editable mode).
- ui/ → UI logic (chatbot frontend or Gradio-based user interface).
- util/ → Helper functions, utilities, and shared code modules.

⚙️ Configuration Files

- .env → Environment variables (API keys, secrets, configs).
- pyproject.toml → Defines project metadata, dependencies, and build system.



Improvement Recommendations

1. Infrastructure Layer

- Scalability: Introduce container orchestration (Kubernetes, ECS, AKS, GKE) for dialog services and ChatManager.
- Caching: Add a caching layer (Redis/Memcached) for faster product lookups and repeated queries.
- Resilience: Use message queues (RabbitMQ, Kafka, SQS) to decouple Dialog Manager ↔ ChatManager calls.

2. Security, Logging, Monitoring

- Security: Enforce TLS everywhere, implement JWT/OAuth for authentication.
- Observability: Add distributed tracing (OpenTelemetry + Jaeger) to monitor request flows across layers.
- Error Tracking: Integrate monitoring tools (Prometheus + Grafana, ELK stack, Sentry) for real-time alerts.

Improvement Recommendations

3. Dialog Manager Enhancements

- Context Management: Add conversation memory (session store or vector DB for context retrieval).
- Personalization: Connect with a user profile DB to adapt recommendations.
- Fallbacks: Implement fallback handlers when intent is unclear or API calls fail.

4. ChatManager Improvements

- Validation Layer Expansion: Add schema validation (Pydantic/Marshmallow) for structured responses.
- Multi-Source Aggregation: Allow ChatManager to pull data from multiple product databases or external APIs.

Improvement Recommendations

5. API & External Integration

- API Gateway: Add an API gateway layer (Kong, APIM, AWS API Gateway) for centralized routing, throttling, and security.
- Extensibility: Standardize APIs so new recommendation engines can be plugged in easily.
- Third-party Integrations: Integrate with CRM/ERP to support business workflows.

6. UI & User Experience

- Multichannel Support: Extend UI beyond chatbot (mobile app, voice assistant, WhatsApp, web widget).
- Feedback Loop: Collect user feedback on recommendations to retrain/improve models.
- A/B Testing: Experiment with dialog strategies and recommendation algorithms.

Technology Used

python = ">=3.9,<4.0"
fastapi = "^0.111.0"
uvicorn = "^0.30.0"
"pandas",
"fastapi",
"uvicorn[standard]",
"sqlalchemy",
"pydantic",
"alembic",
"python-dotenv",
"gradio",
"tenacity",
"openai",
"pydantic",
"httpx",
"fastapi",
"IPython",
"logging"

Thank you